Finesse Agent Login Trace mithilfe von Protokollen

Inhalt

[Einführung](#)
[Voraussetzungen](#)
[Anforderungen](#)
[Verwendete Komponenten](#)
[Agent Desktop starten](#)
[Anmeldeinformationen des Agenten](#)
[SystemInfo](#)
[API_REQUEST](#)
[Herstellen der BOSH-Verbindung](#)
[Agent-Anmeldung](#)
[Anmeldung durchführen](#)
[Abmeldecodes, Ursachencodes, Telefonbuch](#)

Einführung

Dieses Dokument beschreibt den Prozess für die Anmeldung eines Agenten über das Finesse-System mit den Protokolldateien. Es ist wichtig, den Nachrichtenfluss zwischen den verschiedenen Finesse Components, dem Computer Telefony Integration Server (CTI) und dem Client-Desktop zu verstehen, damit Sie Probleme erfolgreich beheben können.

Voraussetzungen

Anforderungen

Cisco empfiehlt, die Befehlszeilenaufforderung für Cisco Finesse und die VOS-CLI (Voice Operating System) zu kennen.

Verwendete Komponenten

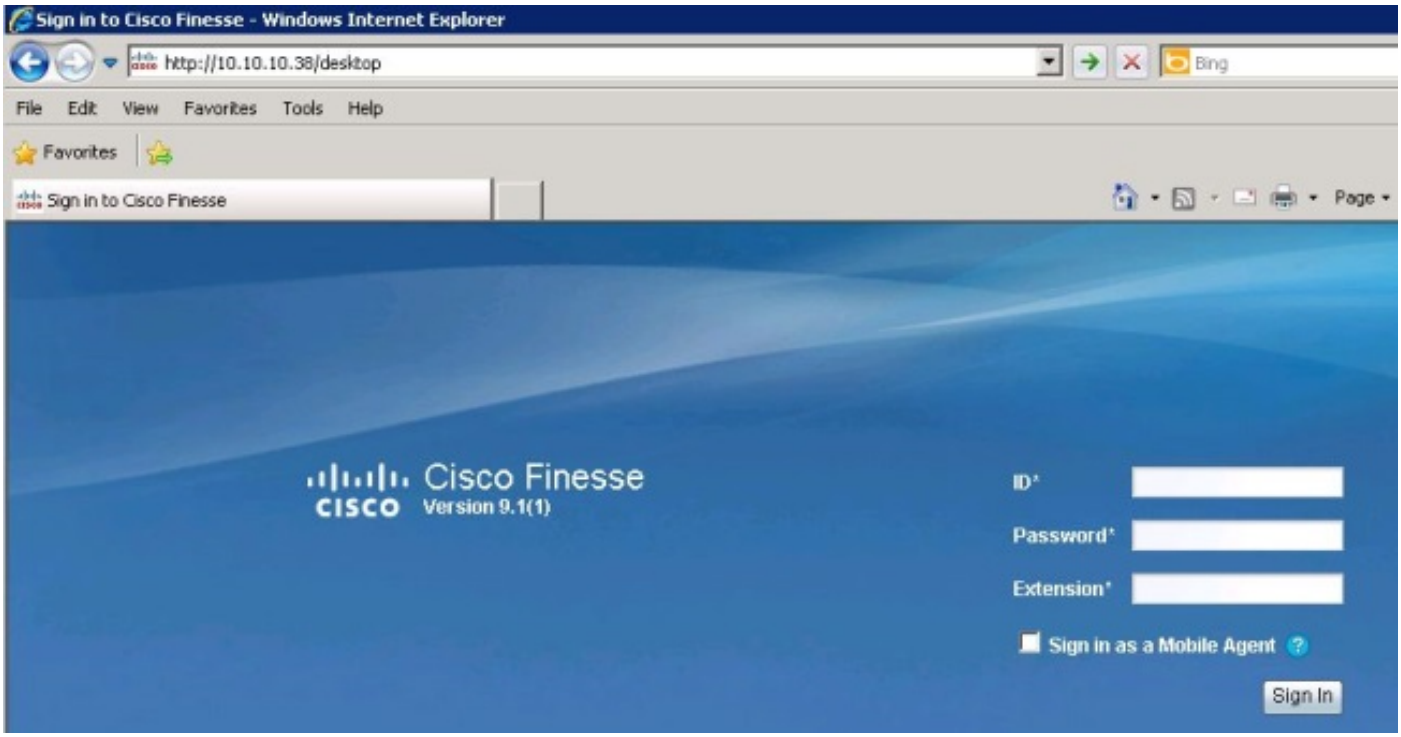
Die Informationen in diesem Dokument basieren auf Cisco Finesse Version 9.1(1).

Die Informationen in diesem Dokument wurden von den Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie

die potenziellen Auswirkungen eines Befehls verstehen.

Agent Desktop starten

Kopieren Sie diese URL in den Webbrowser, um den Agent-Desktop zu starten: **http://<Ihr Finesse-Server>/Desktop**. In Finesse Version 9.1 wird HTTP oder HTTPS unterstützt.



Finesse verwendet Tomcat als Webserver. Wenn Sie Ihren Webbrowser starten, wird die Anfrage an Finesse gesendet, um Ihnen den Mitarbeiter-Desktop vorzustellen. Der Befehl Cisco Tomcat **localwise_access_log** zeigt die Anforderung, den Agent-Desktop zu laden.

```
10.10.10.211 10.10.10.211 - - 80 GET /  HTTP/1.1 302 - 141
10.10.10.211 10.10.10.211 - - 80 GET /desktop/container/ HTTP/1.1 200 4541 185
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/css/base.css
HTTP/1.1 200 3093 7
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/css/login.css
HTTP/1.1 200 2185 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/js/Logon.js HTTP/1.1 200 1745 1
10.10.10.211 10.10.10.211 - - 80 GET /desktop/js/utilities/Cookies.js HTTP/1.1
200 2390 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/thirdparty/jquery/js/jquery.tools.
min.js HTTP/1.1 200 15699 1
10.10.10.211 10.10.10.211 - - 80 GET /desktop/thirdparty/jquery/js/jquery-1.5.
min.js HTTP/1.1 200 84523 7
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/sprite_
buttons.png HTTP/1.1 200 3297 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/help.png
HTTP/1.1 200 830 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/cisco_logo.
png HTTP/1.1 200 760 0 200 2205 1
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/bg.jpg
HTTP/ 1.1 200 32222 4
```

Anmeldeinformationen des Agenten

Nachdem der Agent-Desktop angezeigt wurde, geben Sie Ihre Anmeldeinformationen ein. Bevor Finesse die Anmeldeanforderung an den CTI-Server senden kann, muss der Client eine Bidirectional-Streams Over Synchronous HTTP (BOSH)-Verbindung einrichten. Um die BOSH-Verbindung herzustellen, fragt der Client zunächst vom Finesse Server Systeminformationen ab.

SystemInfo

Der Client-Desktop hat eine Representational State Transfer (REST) Application Programming Interface (API)-Anfrage an diese URL gesendet: **/finesse/api/SystemInfo**. Beachten Sie den **nocache=**. Diese eindeutige ID wird verwendet, um diese Anfrage durch das System zu verfolgen. **Rückgesendet mit status=200** zeigt an, dass die Anfrage erfolgreich empfangen wurde.

```
Container : [ClientServices] SystemInfo: requestId='undefined', Making REST
request: method=GET, url='/finesse/api/SystemInfo?nocache=1366756802163'18:40:03:
Container : [ClientServices] SystemInfo: requestId='undefined', Returned
with status=200
```

Wenn Sie keine Clientprotokolle haben, die Anforderung jedoch nachverfolgen müssen, können Sie das Tomcat **localhost_access_log** durchsuchen, um zu bestimmen, wann die REST API-Anforderung erstellt wurde, und um den eindeutigen Bezeichner zu suchen.

```
127.0.0.1 127.0.0.1 - - 80 GET /finesse/api/SystemInfo ?nocache=1366756802163
HTTP/1.1 200 336 120 10.10.10.211 10.10.10.211 2001 - 80 GET /gadgets/makeRequest
?refresh=3600&url=http%3A%2F%2Flocalhost%2Ffinesse%2Fapi%2FSystemInfo%3Fnocache%
3D1366756802163&httpMethod=GET&headers=Authorization%3DBasic%2520MjAwMToyMDAx%
26locale%3Den_US&postData=&authz=&st=&contentType=TEXT&numEntries=3&getSummaries
=false&signOwner=true&signViewer=true&gadget=undefined&container=default&
bypassSpecCache=&getFullHeaders=false HTTP/1.1 200 659 596
```

API_REQUEST

Tomcat sendet diese API-Anforderung an das Finesse REST API Web Application Repository (WAR). Um die Finesse REST API-Protokolle zu finden, durchsuchen Sie das Finesse Webservices-Protokoll entweder nach dem Timestamp oder der nocache-ID, um API_REQUEST zu finden. Dieses Protokoll zeigt die **REQUEST_START**, die **REQUEST_URL**, **REQUEST_END** und die **verstrichene_Zeit**, die das System zum Abschließen der Anforderung benötigt hat.

```
%CCBU_http-8080-7-6-REQUEST_START: %[method_name=GET][parameter_name=
{ nocache=[1366756802163],  }][resource_name=/SystemInfo][usr=]: Request start
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=SystemInfo][agent_id=][request_
identifier=][request_method=systemInfo.GET][request_parameters=]: Request from
client to webservice api
%CCBU_http-8080-7-6-REGISTER_API_STATS_OBJECT: %[resource_name=com.cisco.ccbu:
category=WebAppStats,component0=SystemInfo-GET]: Registered new api stats object
for new request type. %CCBU_http-8080-7-6-REQUEST_END: %[elapsed_time=98]: Request complete
```

Hier wird der Inhalt angezeigt, der an den Client durch die REST API-Anforderung zum Abrufen von Systeminformationen zurückgegeben wird. Diese Informationen befinden sich in den Client-(Agent-)Protokollen.

```
content='<SystemInfo>
<primaryNode>
<host>UCCEFINESSE91.vmload.cvp</host>
</primaryNode>
<secondaryNode>
<host>UCCEFINESSE138.vmload.cvp</host>
</secondaryNode>
<status>IN_SERVICE</status>
<xmppDomain>UCCEFINESSE138.vmload.cvp</xmppDomain>
<xmppPubSubDomain>pubsub.UCCEFINESSE138.vmload.cvp</xmppPubSubDomain>
</SystemInfo>'
```

Herstellen der BOSH-Verbindung

Die SystemInfo zeigt die primären und sekundären Finesse-Server, den Status von Finesse als **IN_SERVICE**, die **xmppDomain** und die **xmppPubSubDomain an**. Der Client verfügt jetzt über genügend Informationen, um eine BOSH-Verbindung herzustellen.

```
18:40:03: Container : PageServices.init().onLoad: System info status: IN_SERVICE
18:40:03: Container : PageServices.init(): Establishing BOSH connection...
18:40:03: Container : PageServices.init(): Starting timeout and poller...
18:40:04: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: connecting
18:40:04: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: disconnected
18:40:04: Container : PageServices._onDisconnect(): retryCount=0, retrying...
18:40:04: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: connecting
18:40:05: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: connected
18:40:05: Container : PageServices.onLoad(): BOSH established!
```

Der Client wird erfolgreich zum Finesse Object (knoten) **/finesse/api/user/2001** abonniert, sobald die BOSH-Verbindung hergestellt ist.

Wenn die BOSH-Verbindung des Clients hergestellt ist, erhält das Webdienstprotokoll eine **PRESENCE_NOTIFICATION-**Nachricht vom Client. Dieser **PRESENCE_TYPE** gibt nur an, dass der Client für den Empfang von **XMPP-Ereignissen** verfügbar ist und nichts mit der Verfügbarkeit von Agenten in Unified Contact Center Enterprise (UCCE) zu tun hat. Denken Sie daran, dass der Agent noch nicht angemeldet ist.

> **Hinweis**: Die **PRESENCE_TYPE-**Meldungen werden nur angezeigt, wenn ein Client eine BOSH-Verbindung herstellt oder wenn die BOSH-Verbindung eines Clients getrennt wird. Wenn die BOSH-Verbindung des Clients getrennt wird, wird **PRESENCE_TYPE** als nicht verfügbar angezeigt.

Hier ist die Benachrichtigungsereignis im Webservices-Protokoll:

```
%CCBU_Smack Listener Processor (1)-6-PRESENCE_NOTIFICATION_RECIEVED:
%[FROM JID=2001@uccefinesse138.vmload.cvp/desktop]
[PRESENCE_TYPE=available]: Finesse received a presence notifcation
```

Agent-Anmeldung

Nachdem der Client die BOSH-Verbindung hergestellt hat, beginnt der Anmeldeprozess. Der Client stellt eine weitere REST-API-Anfrage, um aktuelle Benutzerinformationen abzurufen. Um diese Anforderung zu stellen, navigieren Sie zu folgender URL: **/finesse/api/user/2001** und enter method=**GET**.

Da es sich um eine andere API-Anforderung handelt, ist die **nocache-ID** anders. Um diese Anfrage verfolgen zu können, müssen Sie diese neue ID verwenden.

```
Container : PageServices.onLoad(): BOSH established! Commencing sign-in process
Container : [ClientServices] User: requestId='undefined', Making REST request:
method=GET, url='/finesse/api/User/2001?nocache=1366756805180
'18:40:05: Container : [ClientServices] User: requestId='undefined',
Returned with status=200,
```

Diese Anforderung finden Sie bei Bedarf im Tomcat **localhost_access_log**. So finden Sie es im Webservices-Protokoll:

```
%CCBU_http-8080-7-6-REQUEST_START: %[method_name=GET][parameter_name={ nocache=
[1366756805180],  }][resource_name=/User/2001][usr=2001]: Request start
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=User/2001][agent_id=2001]
[request_identifier=null][request_method=user.GET][request_parameters=2001]:
Request from client to webservice api
```

Die Anforderung finden Sie im Protokoll "Notification Services" (Benachrichtigungsdienste). Beachten Sie **HTTP/1.1 200 ok**.

> **Hinweis**: Das Cisco Benachrichtigungsprotokoll dient nur zu Informationszwecken. Wenn Sie die Cisco Finesse Notification-Protokollierung aktivieren, wirkt sich dies auf die Leistung aus.

```
>> "GET /finesse/api/User/2001 HTTP/1.1[\r][\n]"
Adding Host request header
>>"Authorization: Basic MjAwMToyMDAx[\r][\n]"
>>"User-Agent: Jakarta Commons-HttpClient/3.1[\r][\n]"
>>"Host: localhost:8080[\r][\n]"
>>"[\r][\n]"
<<"HTTP/1.1 200 OK[\r][\n]"
<<"HTTP/1.1 200 OK[\r][\n]"
<<"Pragma: No-cache[\r][\n]"
<<"Cache-Control: no-cache[\r][\n]"
```

Nachdem der Benachrichtigungsdienst die Anforderung hat, veröffentlicht er die Informationen für diesen Benutzer. Hier ist der POST-Test aus dem Benachrichtigungsdienstprotokoll, das zum Client geleitet wird:

```
Cookie accepted: "$Version=0; JSESSIONIDSSO=C11F62C59D0D0438CDEDEEB0DB12AA0B;
$Path=/"
Cookie accepted: "$Version=0; JSESSIONID=25FE81BD7DB73280A07B4CA4138E7680;
$Path=/finesse"
Buffering response body
<<"<User>[\n]"
<<" <dialogs>/finesse/api/User/2001/Dialogs</dialogs>[\n]"
<<" <extension></extension>[\n]"
<<" <firstName>Mickey</firstName>[\n]"
<<" <lastName>Mouse</lastName>[\n]"
<<" <loginId>2001</loginId>[\n]"
<<" <loginName>mmouse</loginName>[\n]"
```

```
<<" <roles>[\n]"
<<" <role>Agent</role>[\n]"
<<" </roles>[\n]"
<<" <state>LOGOUT</state>[\n]"
<<" <stateChangeTime></stateChangeTime>[\n]"
<<" <teamId>5000</teamId>[\n]"
<<" <teamName>Minnies_Team</teamName>[\n]"
<<" <uri>/finesse/api/User/2001</uri>[\n]"
<<"</User>"
```

Dieses **XMPP-Ereignis**, das in diesem Beispiel **Agent 2001** ist, wird an alle Abonnement-Clients gesendet. Das JavaScript des Clients empfängt das **XMPP Event**, und das Event wird innerhalb des Clients an das Gadget gesendet. Die folgenden Clientprotokolle zeigen den Inhalt der Antwort:

```
Commencing sign-in process18:40:05: Container : [ClientServices] User: requestId=
'undefined', Maurl='/finesse/api/User/2001?nocache=1366756805180'18:40:05:
Container : [ClientServices] User: requestId='undefined', Returned with status=200,
content='<User> king REST request: method=GET,
<dialogs>/finesse/api/User/2001/Dialogs</dialogs>
<extension></extension>
<firstName>Mickey</firstName>
<lastName>Mouse</lastName>
<loginId>2001</loginId>
<loginName>mmouse</loginName>
<roles>
<role>Agent</role>
</roles>
<state>LOGOUT</state>
<stateChangeTime></stateChangeTime>
<teamId>5000</teamId>
<teamName>Minnies_Team</teamName>
<uri>/finesse/api/User/2001</uri>
</User>
```

Anmeldung durchführen

Der Client ist nun bereit, die Anmeldung durchzuführen. Beachten Sie die **RequestID**. Die RequestID wird im Text der Anfrage gesendet. Sie verwenden diese RequestID, um der Anmeldeanforderung an die **REST-API > CTI > REST API > Notification Service > Response** zurück an den Client zu folgen. Diese Anforderung ist ein PUT, d. h. der Client fordert ein UPDATE oder eine Änderung seines aktuellen Status an.

```
Container : SignIn.handleUserLoad(): Performing login: extn=2003 18:40:05:
Container : [ClientServices] User: requestId='6e210ca9-5786-43bc-babf-
64a397a6057f',
</data>
<event>PUT</event>
<requestId>6e210ca9-5786-43bc-babf-64a397a6057f</requestId>
<source>/finesse/api/User/2001</source>
</Update>
```

Die Finesse REST API erhält diese Anfrage vom Client. Anschließend sendet die API einen **SetAgentStateReq** an den CTI-Server.

```
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=User/2001][agent_id=2001]
[request_identifier=6e210ca9-5786-43bc-babf-64a397a6057f][request_method=
```

```
user.PUT][request_parameters= extension:2003 state:LOGIN]: Request from
client to webservice api
%CCBU_http-8080-7-6-REGISTER_API_STATS_OBJECT: %[resource_name=com.cisco.ccbu:
category=WebAppStats,component0=User-[id]-PUT]: Registered new api stats object
for new request type.
%CCBU_http-8080-7-6-REQUEST_END: %[elapsed_time=8]: Request complete
%CCBU_pool-5-thread-4-6-MESSAGE_TO_CTI_SERVER: %[cti_message=Invoke id :20 ,
agentstate : 0, workmode : 0, reason code: -15532, forceflag :1, agentcapacity:
0, agentext: 2003, agentid: 2001][cti_message_name=SetAgentStateReq]:
Message going to the backend cti server
```

Der CTI-Server erhält die Anforderung.

```
Trace: AGENT_EVENT: ID=2001 Periph=5001 Ext=2003 Inst=2003 Sig=
Trace: SkgState=LOGIN SkgDuration=0 OverallState=NOT_READY OverallDuration=0
Reason=0
Trace: MRDID=1 NumTasks=0 MaxTaskLimit=1 AgtMode=1 AGTAvailabilityStatus=0
ICMAgtID=5001
Trace: SkTgtID=5001 SkGrpNo=0x0 SkGrpID=5006 NumLines=0 CurLine=0 ClientStatus=
0x0 Direction=0
```

Wenn der Agent mit dem Status **NOT_READY** angemeldet ist, sendet der CTI-Server **AGENT_STATE-EVENT** an Finesse.

```
MsgType:AGENT_STATE_EVENT (MonitorID:0 PeripheralID:5001 SessionID:0x0
PeripheralType:EnterpriseAgent SkillGroupState:LOGIN StateDuration:0
SkillGroupNumber:85881 SkillGroupID:5000 SkillGroupPriority:0 AgentState:
NOT_READY EventReasonCode:0 MRDID:1 NumTasks:0 AgentMode:1 MaxTaskLimit:1
ICMAgentID:5001 AgentAvailabilityStatus:0 NumFltSkillGroups:0 Direction:0
ClientSignature:""AgentID:"2001" AgentExtension:"2003" AgentInstrument:"2003"
RemaskNumMasks:1 RemaskInstrument:"2003" RemaskExtension:"2003" RemaskCallId:
0xffffffff RemaskFunctionFlag:<0x38> <LogoutCodeReq,NotRdyCodeReq,WrapDataReq>
RemaskCallMask:<0x21000000> <MC,Emerg> RemaskAgentMask:<0x0a000000> <
Logout,Avail> )Trace: AGENT_EVENT: ID=2001 Periph=5001 Ext=2003 Inst=2003 Sig=
Trace: SkgState=LOGIN SkgDuration=0 OverallState=NOT_READY OverallDuration=0
Reason=0 Trace: MRDID=1 NumTasks=0 MaxTaskLimit=1 AgtMode=1
AGTAvailabilityStatus=0 ICMAgtID=5001
```

Dies ist das Webservice-Protokoll, das die Veranstaltung vom CTI-Server erhalten hat. Denken Sie daran, dass Sie die Meldung **RAW** zuerst vom CTI-Server sehen und dann die Meldung **Decoded (Entschlüsselt)** sehen.

```
%CCBU_CTIMessageEventExecutor-0-6-DECODED_MESSAGE_FROM_CTI_SERVER: %[cti_message
=CTIAgentStateEvent [skillGroupState=0 (LOGIN), stateDuration=0, skillGroupNumber
=85881, skillGroupPriority=0, agentState=2 (NOT_READY), eventReasonCode=0,
numFltSkillGroups=0,CTIClientSignature=, agentID=2001, agentExtension=2003,
agentInstrument=2003, agentID_ Long=null, duration=null, nextAgentState=null,
fltSkillGroupNumberList=[], fltSkill GroupIDList=[], fltSkillGroupPriorityList=[],
fltSkillGroupStateList=[]]CTIMessageBean [invokeID=null, msgID=30, timeTracker=
{"id":"AgentStateEvent","CTI_MSG_RECEIVED":1366756808374,
"CTI_MSG_DISPATCH":1366756808375}, msgName=AgentStateEvent, deploymentType=CCE]]
[cti_response_time=1]: Decoded Message to Finesse from backend cti server
```

Nachdem Finesse das **AgentStateEvent** vom CTI-Server erhalten hat, muss das Ereignis an den Benachrichtigungsdienst **veröffentlicht** werden, damit der Client das UPDATE erhält. Der Agent kann nur durch das Empfangen dieses **XMPP-Ereignisses** wissen, dass sich sein Status geändert hat. Finesse konvertiert das **AgentStateEvent** in **XMPP** und sendet das **XMPP** an den Benachrichtigungsdienst. Beachten Sie, dass das Ereignis ein **PUT** ist, und die RequestID in der Payload ist.

%CCBU_pool-5-thread-5-6-**XMPP_PUBLISH_ASYNCHRONOUS**: %[NodeId=**/finesse/api/User/2001**][**Payload**=<Update><data><user><dialogs>/finesse/api/User/2001/Dialogs</dialogs><extension>2003</extension><firstName>Mickey</firstName><lastName>Mouse</lastName><loginId>2001</loginId><loginName>mmouse</loginName><reasonCodeId>-1</reasonCodeId><roles><role>Agent</role></roles><state>NOT_READY</state><stateChangeTime>2013-04-23T22:40:08Z</stateChangeTime><teamId>5000</teamId><teamName>Minnies_Team</teamName><uri>/finesse/api/User/2001</uri></user></data><event>PUT</event><requestId>**6e210ca9-5786-43bc-babf-64a397a6057f** </requestId><source>/finesse/api/User/2001</source></Update>]: **Publishing XMPP Message Asynchronously**

Hier erhält der Benachrichtigungsdienst das UPDATE. Obwohl in der Meldung angegeben wird, dass **das Paket nicht an JID weitergeleitet wurde**, wird dem Benutzer eine Meldung gesendet, dass ein Ereignis veröffentlicht wurde.

RoutingTableImpl: Failed to route packet to **JID: 2001@uccefinesse138.vmload.cvp/User** packet: <message from="pubsub.uccefinesse138.vmload.cvp" to="2001@uccefinesse138.vmload.cvp/ User" id="/finesse/api/User/2001__2001@uccefinesse138.vmload.cvp__VI1B2"><event xmlns="http://jabber.org/protocol/pubsub#event"><items node="/finesse/api/User/2001"><item id="1su0Keff8M2irdS"><notification xmlns="http://jabber.org/protocol/pubsub">
<Update>

Der Text der Nachricht lautet wie folgt:

<data>
<user>
<dialogs>/finesse/api/User/2001/Dialogs</dialogs>
<extension>2003</extension>
<firstName>Mickey</firstName>
<lastName>Mouse</lastName>
<loginId>2001</loginId>
<loginName>mmouse</loginName>
<reasonCodeId>-1</reasonCodeId>
<roles>
<role>Agent</role>
</roles>
<state>NOT_READY</state>
<stateChangeTime>2013-04-23T22:40:08Z</stateChangeTime>
<teamId>5000</teamId>
<teamName>Minnies_Team</teamName>
<uri>/finesse/api/User/2001</uri>
</user>
</data>
<event>**PUT**</event>
<requestId>**6e210ca9-5786-43bc-babf-64a397a6057f<**;/requestId>
<source>/finesse/api/User/2001</source>
</Update></notification></item></items></event></message>

Wie zuvor wird die XMPP-Nachricht vom Client empfangen und an das Gadget des Clients übermittelt. Beachten Sie, dass der Client das Ereignis mit der ursprünglichen RequestID in der Nachricht empfängt.

Returned with status=202, content=''18:40:05: Container : [ClientServices] MasterPublisher._eventHandler() - Received event on node '/finesse/api/User/2001': <Update>
<data>
<user>
<dialogs>/finesse/api/User/2001/Dialogs</dialogs>

```
<extension>2003</extension>
<firstName>Mickey</firstName>
<lastName>Mouse</lastName>
<loginId>2001</loginId>
<loginName>mmouse</loginName>
<reasonCodeId>-1</reasonCodeId>
<roles>
<role>Agent</role>
</roles>
<state>NOT_READY</state>
<stateChangeTime>2013-04-23T22:40:08Z</stateChangeTime>
<teamId>5000</teamId>
<teamName>Minnies_Team</teamName>
<uri>/finesse/api/User/2001</uri>
</user>
</data>
<event>PUT</event>
<requestId>6e210ca9-5786-43bc-babf-64a397a6057f</requestId>
<source>/finesse/api/User/2001</source>
</Update>
```

Jetzt ist der Client erfolgreich angemeldet.

```
Container : SignIn._triggerLoggedIn(): Successfully logged in!18:40:05
```

Abmeldecodes, Ursachencodes, Telefonbuch

Nun muss der Client agentenspezifische Daten abrufen, z. B. Abmeldecodes, Ursachencodes und Telefonbuch. Hier ist die Anfrage für diese Informationen an den Kunden.

```
Container : SignIn._triggerLoggedIn(): Successfully logged in!18:40:05:
Container : [ClientServices] Dialogs: requestId='undefined', Making REST
request: method=GET, url='/finesse/api/User/2001/Dialogs?nocache=
1366756805355?

18:40:05: Container : [ClientServices] User: requestId='undefined',
Making REST request: method=GET, url='/finesse/api/User/2001/ReasonCodes?
category=LOGOUT&nocache=1366756805356'18:40:05: Container : [ClientServices]
User: requestId='undefined', POST_DATA=''18:40:05: Container : _displayUserData
(): User's current state is: NOT_READY

'18:40:05: Container : [ClientServices] User: requestId='undefined', Making REST
request: method=GET, url='/finesse/api/User/2001/ReasonCodes?category=NOT_READY&
nocache=1366756805358

18:40:05: Container : [ClientServices] User: requestId='undefined', POST_DATA=
''18:40:05: Header : The client logger has been initialize for the header
18:40:05: Header : _displayUserData(): User's current state is: NOT_READY

18:40:05: Header : Container._initGadgetContainer(): Initializing gadget
container.
18:40:05: Header : FailoverMonitor.startListening(): Listening for triggers
18:40:05: Header : PageServices.stopTimeoutPoller(): Cancelling connection
timeout and poller...
18:40:05: Header : [ClientServices] id=2001: TypeError: 'this._listenerCallback
[...].callback' is null or not an object
```

Dieselbe Logik gilt für diese Anforderungen. Beachten Sie, dass die Finesse-Ursachencodes und das Telefonbuch in der Finesse-Datenbank und nicht in UCCE gespeichert sind.